AF/2122 IFW

PTO/SB/21 (09-04)
Approved for use through 07/31/2006. OMB 0651-0031
U.S. Patent and Trademark Office; U.S. DEPARTMENT OF COMMERCE

TRANSMITTAL FORM

(to be used for all correspondence after initial filing)

Total Number of Pages in This Submission: 31

Application Number	10/020,697
Filing Date	10/29/2001
First Named Inventor	Cashin
Art Unit	2122
Examiner Name	Vo, Ted T.
Attorney Docket Number	RPS920010135US1

ENCLOSURES *(Check all that apply)*

- ☐ Fee Transmittal Form
  - ☐ Fee Attached
- ☐ Amendment/Reply
  - ☐ After Final
  - ☐ Affidavits/declaration(s)
- ☐ Extension of Time Request
- ☐ Express Abandonment Request
- ☐ Information Disclosure Statement
- ☐ Certified Copy of Priority Document(s)
- ☐ Reply to Missing Parts/ Incomplete Application
  - ☐ Reply to Missing Parts under 37 CFR 1.52 or 1.53
- ☐ Drawing(s)
- ☐ Licensing-related Papers
- ☐ Petition
- ☐ Petition to Convert to a Provisional Application
- ☐ Power of Attorney, Revocation Change of Correspondence Address
- ☐ Terminal Disclaimer
- ☐ Request for Refund
- ☐ CD, Number of CD(s) _____
  - ☐ Landscape Table on CD
- ☐ After Allowance Communication to TC
- ☐ Appeal Communication to Board of Appeals and Interferences
- ☑ Appeal Communication to TC **(Appeal Notice, Brief, Reply Brief)**
- ☐ Proprietary Information
- ☐ Status Letter
- ☑ Other Enclosure(s) (please Identify below):
  Return Postcard

Remarks

Appellants submit herewith an Amended Appeal Brief that clarifies the Real Party in Interest.

SIGNATURE OF APPLICANT, ATTORNEY, OR AGENT

Firm Name	Winstead Sechrest & Minick P.C.		
Signature			
Printed name	Robert A. Voigt, Jr.		
Date	08/15./2005	Reg. No.	47,159

CERTIFICATE OF TRANSMISSION/MAILING

I hereby certify that this correspondence is being facsimile transmitted to the USPTO or deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to: Commissioner for Patents, P.O. Box 1450, Alexandria, VA 22313-1450 on the date shown below:

Signature	Toni Stanley		
Typed or printed name	Toni Stanley	Date	08/15/2005

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In re Application of: Cashin et al.	:	Before the Examiner: Vo, Ted T.
Serial No.: 10/020,697	:	Group Art Unit: 2122
Filing Date: October 29, 2001	:	
Title: OBJECT ORIENTED MODEL OF PRELOADING SOFTWARE	:	Lenovo, Inc. P.O. Box 12195 Dept. 9CCA, Bldg. 002-2 Research Triangle Park, NC 27709

AMENDED APPEAL BRIEF

Mail Stop Appeal Brief-Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

I. REAL PARTY IN INTEREST

At the time of the filing of the above-identified application, International Business Machines Corporation was the assignee of the entire right, title and interest in the above-identified patent application. International Business Machines Corporation is currently

CERTIFICATION UNDER 37 C.F.R. §1.8

I hereby certify that this correspondence is being deposited with the United States Postal Service with sufficient postage as first class mail in an envelope addressed to Mail Stop Appeal Brief-Patents, Commissioner for Patents, P.O. Box 1450, Alexandria, Virginia 22313-1450, on August 15, 2005.

Toni Stanley
Signature

Toni Stanley
(Printed name of person certifying)

undergoing negotiations with Lenovo (Singapore) Pte. Ltd. to transfer some of its intellectual property to them, including assigning the above-identified patent application to Lenovo (Singapore) Pte. Ltd.

II. RELATED APPEALS AND INTERFERENCES

There are no other appeals or interferences known to Appellants, Appellants' legal representative or assignee which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1-15 are pending in the Application. Claims 5, 10 and 15 are allowed. Claims 1-4, 6-9 and 11-14 stand rejected. Claims 1-4, 6-9 and 11-14 are appealed.

IV. STATUS OF AMENDMENTS

The Appellants' response to the Office Action having a mailing date of August 26, 2004, has been considered, but the Examiner indicated that it did not place the application in condition for allowance because Appellants' arguments were deemed unpersuasive.

V. SUMMARY OF CLAIMED SUBJECT MATTER

In one embodiment of the present invention, a method for creating a preload, where an object of the preload is an aggregation of one or more software element objects, may comprise the step of defining a particular preload object with one or more attributes. Specification, page 11, lines 13-16; Figure 3, step 301. The method may further comprise comparing attributes of one or more software element objects with one or more attributes of the particular preload object, where each of the one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object. Specification, page 11, lines 17-22; Figure 3, step 302. The method may further comprise identifying one or more of the one or more software element objects whose attributes comprise the one or more attributes of the particular preload object. Specification, page 11, lines 23-

25; Figure 3, step 303. The method may further comprise installing software associated with the identified one or more software elements objects onto a particular preload associated with the particular preload object. Specification, page 14, line 19 – page 15, line 2; Figure 3, step 310.

In another embodiment of the present invention, a computer program product having a computer readable medium having computer program logic recorded thereon for creating a preload may comprise the programming step of defining a particular preload object with one or more attributes. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 13-16; Figure 1, elements 20 and 50; Figure 3, step 301. The computer program product may further comprise comparing attributes of the one or more software element objects with the one or more attributes of the particular preload object, where each of the one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 17-22; Figure 1, elements 20 and 50; Figure 3, step 302. The computer program product may further comprise identifying one or more of the one or more software element objects whose attributes comprise the one or more attributes of the particular preload object. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 23-25; Figure 1, elements 20 and 50; Figure 3, step 303. The computer program product may further comprise installing software associated with the identified one or more software elements objects onto a particular preload associated with the particular preload object. Specification, page 7, line 3 – page 8, line 17; Specification, page 14, line 19 – page 15, line 2; Figure 1, elements 20 and 50; Figure 3, step 310.

In another embodiment of the present invention, a system may comprise a processor. Figure 1, element 10. The system may further comprise a memory unit coupled to the processor, where the memory unit is operable for storing a computer

program for creating a preload, where an object of the preload is an aggregation of one or more software element objects. Specification, page 7, line 3 – page 8, line 17; Specification, page 9, line 24 – page 10, line 3; Figure 1, elements 10, 14 and 50. The computer program may be operable for performing the programming step of defining a particular preload object with one or more attributes. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 13-16; Figure 1, elements 20 and 50; Figure 3, step 301. The computer program may further be operable for performing the programming step of comparing attributes of the one or more software element objects with the one or more attributes of the particular preload object, where each of the one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 17-22; Figure 1, elements 20 and 50; Figure 3, step 302. The computer program may further be operable for performing the programming step of identifying one or more of the one or more software element objects whose attributes comprise the one or more attributes of the particular preload object. Specification, page 7, line 3 – page 8, line 17; Specification, page 11, lines 23-25; Figure 1, elements 20 and 50; Figure 3, step 303. The computer program may further be operable for performing the programming step of installing software associated with the identified one or more software element objects onto a particular preload associated with the particular preload object. Specification, page 7, line 3 – page 8, line 17; Specification, page 14, line 19 – page 15, line 2; Figure 1, elements 20 and 50; Figure 3, step 310.

VI. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

Claims 1-4, 6-9 and 11-14 stand rejected under 35 U.S.C. §102(e) as being anticipated by Teoman et al. (U.S. Patent No. 6,370,614) (hereinafter "Teoman").

VII. ARGUMENT

The Examiner has rejected claims 1-4, 6-9 and 11-14 under 35 U.S.C. §102(e) as being anticipated by Teoman. Paper No. 5, page 2. Appellants respectfully traverse these rejections for at least the reasons stated below.

For a claim to be anticipated under 35 U.S.C. §102, each and every claim limitation must be found within the cited prior art reference and arranged as required by the claim. M.P.E.P. §2131.

A. Claims 1, 6 and 11 are not anticipated by Teoman.

Appellants respectfully assert that Teoman does not disclose "comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object" as recited in claim 1 and similarly in claims 6 and 11. The Examiner cites column 14, lines 61-66 of Teoman and Figure 6 of Teoman as disclosing the above-cited claim limitations. Paper No. 5, pages 10-11. Appellants respectfully traverse and assert that Teoman instead discloses data integrity checking which may encompass having the data in the user cache being periodically compared against the corresponding data in the mass storage device. Column 14, lines 61-63. Teoman further discloses that if the data does not match, the operating system or user (or both) may be altered so that an archive of the correct data may be generating. Column 14, lines 63-66. Comparing data in the user cache against the corresponding data in the mass storage data is not the same as comparing attributes of a software element object with attributes of a preload object. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

Furthermore, Figure 6 of Teoman instead discloses file and driver objects that are used, in at least one embodiment, to determine the sequence in which device

drivers are called to service an I/O request. There is no language in the description of Figure 6 of Teoman that discloses comparing attributes of a software element object with attributes of a preload object. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

In response to Appellants' above argument, the Examiner cites column 12, lines 2-4 and 18-21 and column 14, lines 59-63 of Teoman as well as cites elements 15, 45, 105, 108, 110 and 112 of Teoman and cites Figures 6 and 7 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, pages 4-5. Appellants respectfully traverse.

Teoman instead discloses that an observer (element 95) receives an access packet that includes a handle to the file object and values indicating the amount of data to be accessed, the nature of the access and so forth. Column 12, lines 2-5. Teoman further discloses that the user cache driver (element 45) compares the blocks indicated by the access packet to the directory of the user cache (element 25) to determine if the blocks are present in the user cache. Comparing the blocks indicated by an access packet with a directory of a cache is not the same as comparing attributes of a software element object with attributes of a particular preload object. The Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that comparing blocks indicated by the access packet to the directory of the user cache (element 25) to determine if the blocks are present in the user cache is the same as comparing attributes of a software element object with attributes of a particular preload object. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434

(Fed. Cir. 2002).

Furthermore, Teoman instead discloses that because data stored in the user cache corresponds to and should be a duplicate of data stored in a mass storage device, data integrity checking can be implemented. Column 14, lines 59-61. Teoman further discloses that data in the user cache is periodically compared against the corresponding data in the mass storage device. Column 14, lines 62-63. Teoman further discloses that if the data does not match, the operating system or user (or both) may be alerted so that an archive of the correct data may be generated. Column 14, lines 63-66. Hence, Teoman discloses periodically comparing the data in the user cache with corresponding data in the mass storage device. Comparing the data in the user cache with corresponding data in the mass storage device is not the same as comparing attributes of a software element object with attributes of a particular preload object. The Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that comparing the data in the user cache with corresponding data in the mass storage device is the same as comparing attributes of a software element object with attributes of a particular preload object. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Furthermore, the Examiner focuses on "data integrity checking" mentioned in the above paragraph. Paper No. 5, page 4. The Examiner has not clearly provided any connection between "data integrity checking" and "comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an

application software object" as recited in claims 1, 6 and 11. Again, the Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that "data integrity checking" is the same as comparing attributes of a software element object with attributes of a particular preload object. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Furthermore, Teoman instead discloses that each file object (element 105) includes a number of attributes and methods for accessing the attributes. Column 11, lines 4-6. Teoman further discloses that the attributes may include, for example, an offset within the associated file (element 15) at which the next read or write will begin, one or more values that indicate where the file (element 15) is stored such as a logical storage identifier and a directory identifier, and a pointer to a driver object (element 107) ultimately associated with a device driver (element 47). Hence, Teoman discloses a file object that includes a pointer (element 108) that points to a device driver (element 47). There is no language in Teoman that discloses that attributes of a software element object is compared with the attributes of the file object (Examiner asserts that the file object in Teoman discloses the preload object as claimed). The Examiner has not cited to any particular element in Teoman as disclosing such a software element object. If the Examiner is asserting that a device driver object (elements 107, 109 and 111 in Teoman) as disclosed in Teoman is such a software element object, then the Examiner must provide evidence that the attributes of a device driver object (elements 107, 109 and 111 in Teoman) is compared with the attributes of file object (Examiner asserts that file object 105 in Teoman discloses the preload object as claimed). Since the Examiner has not provided such evidence, the Examiner has not presented a *prima facie* case of anticipation in rejecting claims 1, 6

and 11. M.P.E.P. §2131.

In response to Appellants citing *Ex parte Levy* and *In re Robertson* in the previous paragraphs, the Examiner correctly notes that these cases relate to inherency. Paper No. 5, page 5. However, while these cases relate to inherency, Appellants are citing these cases for the inferred proposition that the Examiner must provide a basis in fact and/or technical reasoning to support the Examiner's statements. That is, the Examiner must support his interpretation of a reference with objective evidence or cogent technical reasoning. The Examiner cannot transmogrify the teachings of Teoman in order to support his assertion that Teoman discloses Appellants' claim limitations. That is why the Examiner must support his interpretation of a reference with objective evidence or cogent technical reasoning.

Furthermore, the Examiner cites *In re Schreiber* to support the assertion that Teoman inherently discloses the aspects of Appellants' claimed invention. Paper No. 5, page 5. The Examiner focuses on "instantiation" of a file object (element 105 of Teoman) and integrity checking, as disclosed in Teoman, as evidence that Teoman structurally discloses Appellants' claimed invention and hence inherently discloses Appellants' claimed invention. Paper No. 5, page 5. Appellants respectfully traverse.

"Instantiation", as disclosed in Teoman, refers to "creating" an object, such as a file object. Teoman discloses that when a file (element 15) is first opened for access, the I/O manager instantiates a file object (element 105) in system memory to represent the file and returns a value called a file reference to the application that requested the file to be opened. Column 10, line 63 – column 11, line 3. Hence, Teoman discloses creating a file object in memory to represent a file.

Furthermore, "integrity checking" refers to ensuring that correct data is generated. Teoman discloses that the data in the user cache is periodically compared against the corresponding data in the mass storage device. Column 14, lines 62-63.

Teoman further discloses that if the data does not match, the operating system or user may be alerted so that an archive of the correct data may be generated. Column 14, lines 63-66. Hence, Teoman discloses ensuring that correct data is generated.

The teaching of creating a file object in memory to represent a file as well as the teaching of ensuring that correct data is generated does not structurally disclose Appellants' claimed invention. Appellant has no claim limitations directed to creating a file object or generating correct data. The Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that "data integrity checking" and "instantiation", as disclosed in Teoman, discloses Appellants' claimed invention. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Appellants further assert that Teoman does not disclose "identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object" as recited in claim 1 and similarly in claims 6 and 11. The Examiner cites Figure 6 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, page 11. Appellants respectfully traverse and assert that Figure 6 of Teoman instead discloses file and driver objects that are used, in at least one embodiment, to determine the sequence in which device drivers are called to service an I/O request. There is no language in the Teoman that discloses that these file and driver objects comprise attributes of a preload object. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

In response to Appellants' above argument, the Examiner asserts that the file object (element 105) of Teoman discloses a preload object and that the file (element

15) of Teoman in conjunction with the file object (element 105) of Teoman disclose the above-cited claim limitation. Paper No. 5, page 6. Appellants respectfully traverse.

Teoman instead discloses that each file object (element 105) includes a number of attributes and methods for accessing the attributes. Column 11, lines 4-6. Teoman further discloses that the attributes may include, for example, an offset within the associated file (element 15) at which the next read or write will begin and one or more values that indicate where the file (element 15) is stored such as a logical storage identifier. Column 11, lines 6-9. Hence, Teoman discloses that the file object (element 105) includes attributes which may include a value(s) that indicate where the file (element) is stored. Teoman does not disclose any identification step. Neither does Teoman disclose identifying a software element object. Neither does Teoman disclose identifying a software element object whose attributes comprise the attributes of a particular preload object (Examiner asserts that file object 105 discloses a preload object). The Examiner has not specifically identified in Teoman such a software element object. That is, the Examiner has not identified an element in Teoman which is a software element object whose attributes comprise the attributes of file object 105 (Examiner asserts that file object 105 discloses a preload object). Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

The Examiner continues by citing Figure 11 as support that Teoman discloses the above-cited claim limitation. Paper No. 5, page 6. Appellants respectfully traverse. Teoman instead discloses that Figure 11 is an exemplary user interface generated by the user cache manager to allow the user to specify commanded preloads. Column 2, lines 50-52. There is no language in the description of Figure 11 that discloses identifying a software element objects whose attributes comprise attributes of a particular preload object. Thus, Teoman does not disclose all of the

limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

The Examiner further continues by discussing the data types of files as support that Teoman discloses the above-cited claim limitation. Paper No. 5, page 6. Appellants are confused as to the significance of the Examiner's statements as it relates to the data types of files. Appellants respectfully request the Examiner to more clearly explain the connection between the data types of files and the above-cited claim limitation pursuant to 37 C.F.R. §1.104(c)(2).

Appellants further assert that Teoman does not disclose "installing software associated with said identified one or more software elements objects onto a particular preload associated with said particular preload object" as recited in claim 1 and similarly in claims 6 and 11. The Examiner cites column 15, lines 34-61; column 16, lines 4-7 and Figure 12 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, page 11. Appellants respectfully traverse and assert that Teoman instead discloses examples of preloading policies that include preloading complete files in response to file segment access, preloading all files within the directory or folder of a launched application, preloading all files in a directory or folder if a threshold number of files from the directory or folder have already been accessed, preloading files in the system directory or folder, preloading files having a particular file type identifier if a threshold number of files having the file type identifier have been accessed, and so forth. Column 15, lines 34-43. Teoman further discloses that the user may select a file to indicate that a particular file is to be preloaded into the user cache. Column 16, lines 4-6. Furthermore, Figure 12 of Teoman instead discloses an exemplary user interface generated by the user cache manager to permit the user to generate a report of the user cache contents, test the memory in the user cache, flush the user cache or transfer the contents of the user cache to a mass storage such as a tape backup. While Teoman discloses preloading a particular file by selecting a file associated with the

file to be preloaded, there is no language in Teoman that discloses installing software associated with an identified software element object. Neither is there any language in Teoman that discloses installing software associated with an identified software element object onto a particular preload associated with the particular preload object. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

In response to Appellants' above argument, the Examiner cites column 2, lines 1-10; column 8, lines 65-67; and column 16, lines 1-37 of Teoman as well as Figures 6 and 11 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, pages 6-7. Appellants respectfully traverse.

Teoman instead discloses that it is determined whether data from the storage location is cached in a primary cache maintained in a system memory of the computer system. Column 2, lines 1-4. The Examiner has not provided any basis in fact and/or technical reasoning to support the assertion that caching data is the same as installing software associated with identified software element objects onto a particular preload (Examiner asserts that element 105 discloses a particular preload) associated with the particular preload. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Further, Teoman instead discloses that once data from the remote database is cached in the user cache, the host computer can access the data virtually instantly, avoiding the performance and reliability issues inherent in the network and reducing the overall network traffic. Column 8, line 66 – column 9, line 3. Again, the

Examiner has not provided any basis in fact and/or technical reasoning to support the assertion that caching data is the same as installing software associated with identified software element objects onto a particular preload (Examiner asserts that element 105 discloses a particular preload) associated with the particular preload. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 1, 6 and 11. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Furthermore, Teoman instead discloses that the user may double click selected logical drives, directories or filenames to indicate that files meeting the specified criteria are to be preloaded. Column 16, lines 1-4. While Teoman discloses preloading files that meet specified criteria, there is no language in the cited passage that discloses installing software associated with identified software element objects onto a particular preload (Examiner asserts that element 105 discloses a particular preload) associated with the particular preload. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P. §2131.

Furthermore, Teoman instead discloses that Figure 6 illustrates file and driver objects that are used to determine the sequence in which device drivers are called to service an I/O request. Column 2, lines 33-35. Teoman further discloses that Figure 11 is an exemplary user interface generated by the user cache manager to allow the user to specify commanded preloads. Column 2, lines 50-52. There is no language in the description of Figures 6 and 11 that discloses installing software associated with an identified software element object onto a particular preload associated with the particular preload object. Thus, Teoman does not disclose all of the limitations of claims 1, 6 and 11, and thus Teoman does not anticipate claims 1, 6 and 11. M.P.E.P.

§2131.

B. Claims 2-4, 7-9 and 12-14 are not anticipated by Teoman for at least the reasons stated in Section A.

Claims 2-4 depend from claim 1 and hence are not anticipated by Teoman for at least the reasons that claim 1 is not anticipated by Teoman as discussed above in Section A. Claims 7-9 depend from claim 6 and hence are not anticipated by Teoman for at least the reasons that claim 6 is not anticipated by Teoman as discussed above in Section A. Claims 12-14 depend from claim 11 and hence are not anticipated by Teoman for at least the reasons that claim 11 is not anticipated by Teoman as discussed above in Section A.

C. Claims 2, 7 and 12 are not anticipated by Teoman.

Appellants respectfully assert that Teoman does not disclose "modifying an attribute of said identified one or more software element objects to match said one or more attributes of said particular preload object" as recited in claim 2 and similarly in claims 7 and 12. The Examiner cites Figure 10 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, page 11. Appellants respectfully traverse and assert that Figure 10 of Teoman instead discloses an exemplary user interface generated by the user cache manager to permit the user to configure memory management and preloading policies for the user cache. There is no language in the description of Figure 10 of Teoman that discloses modifying an attribute of a software element object. Neither is there any language in the description of Figure 10 of Teoman that discloses modifying an attribute of a software element object to match one or more attributes of a preload object. Thus, Teoman does not disclose all of the limitations of claims 2, 7 and 12, and thus Teoman does not anticipate claims 2, 7 and 12. M.P.E.P. §2131.

In response to Appellants' above argument, the Examiner cites column 9, lines

65-66; column 14, lines 59-67 as well as Figures 6, 10 and 11 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, page 7. Appellants respectfully traverse.

Teoman instead discloses that the user specifies that all the files in the system directory 'C:\SYSTEM\' are to be preloaded. Column 9, lines 65-66. Specifying the files to be preloaded is not the same as modifying an attribute. Neither is there any language in the passage that discloses modifying an attribute of an identified software element object. Neither is there any language in the passage that discloses modifying an attribute of an identified software element object to match the attributes of a particular preload object (Examiner asserts that element 105 of Teoman discloses a particular preload object). Thus, Teoman does not disclose all of the limitations of claims 2, 7 and 12, and thus Teoman does not anticipate claims 2, 7 and 12. M.P.E.P. §2131.

Furthermore, Teoman instead discloses that because data stored in the user cache corresponds to and should be a duplicate of data stored in a mass storage device, data integrity checking can be implemented. Column 14, lines 59-61. Teoman further discloses that data in the user cache is periodically compared against the corresponding data in the mass storage device. Column 14, lines 62-63. Teoman further discloses that if the data does not match, the operating system or user (or both) may be alerted so that an archive of the correct data may be generated. Column 14, lines 63-66. Hence, Teoman discloses periodically comparing the data in the user cache with corresponding data in the mass storage device. The Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that comparing the data in the user cache with corresponding data in the mass storage device is the same as modifying an attribute of software element objects to match the attributes of a particular preload object (Examiner asserts that element 105 of Teoman discloses a particular preload object). *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464

(Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 2, 7 and 12. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

Teoman further discloses that Figure 6 illustrates file and driver objects that are used to determine the sequence in which device drivers are called to service an I/O request. Column 2, lines 33-35. Teoman further discloses that Figure 10 is an exemplary user interface generated by the user cache manager to permit the user to configure memory management and preloading policies for the user cache. Column 2, lines 47-49. Teoman further discloses that Figure 11 is an exemplary user interface generated by the user cache manager to allow the user to specify commanded preloads. Column 2, lines 50-52. There is no language in the description of Figures 6, 10 and 11 that discloses modifying an attribute of software element objects to match the attributes of a particular preload object (Examiner asserts that element 105 of Teoman discloses a particular preload object). Thus, Teoman does not disclose all of the limitations of claims 2, 7 and 12, and thus Teoman does not anticipate claims 2, 7 and 12. M.P.E.P. §2131.

D. Claims 3, 8 and 13 are not anticipated by Teoman.

Appellants respectfully assert that Teoman does not disclose "wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises one or more of an operating system information and an installation information" as recited in claim 3 and similarly in claims 8 and 13. The Examiner cites column 6, line 56 – column 10, line 27 and Figure 6 of Teoman as disclosing the above-cited claim limitation. Paper No. 5, page 11. Appellants respectfully traverse and assert that Teoman instead discloses processing of I/O

requests. Column 8, line 5 – column 10, line 27. Further, Figure 6 of Teoman instead discloses that file and driver objects are used to determine the sequence in which device drivers are called to service an I/O request. There is no language in the cited passage or in the description of Figure 6 of Teoman that discloses a software element object associated with attribute data where the attribute data includes either operating system information or installation information. Thus, Teoman does not disclose all of the limitations of claims 3, 8 and 13, and thus Teoman does not anticipate claims 3, 8 and 13. M.P.E.P. §2131.

In response to Appellants' above argument, the Examiner cites to M.P.E.P. §2106 (IV)(B)(1)(b), *In re Dembiczak* and to *In re Gulack* asserting that the above-cited limitation is non-functional and then concludes that Teoman discloses operating system information at column 7, line 50 of Teoman and that Teoman discloses installation information at column 9, line 55 – column 10, line 27 of Teoman. Paper No. 5, pages 8-9. Appellants respectfully traverse.

M.P.E.P. §2106 (IV) refers to determining whether the claimed invention complies with 35 U.S.C. §101. In particular, M.P.E.P. §2106 (IV)(B)(1)(b) states that descriptive material that cannot exhibit any functional interrelationship with the way in which computing processes are performed does not constitute a statutory process, machine, manufacture or composition of matter and should be rejected under 35 U.S.C. §101. Appellants are confused as to whether the Examiner is asserting that claims 3, 8 and 13 are directed to non-statutory subject matter or that somehow M.P.E.P. §2106 (IV)(B)(1)(b) supports the assertion that the Examiner does not have to cite a passage in Teoman that discloses that each of the software element objects is associated with attribute data, where the attribute data comprises one or more of an operating system information and an installation information. Appellants respectfully request the Examiner to clarify why the Examiner is citing M.P.E.P. §2106 (IV)(B)(1)(b) pursuant to 37 C.F.R. §1.104(c)(2). Appellants respectfully assert that

claims 3, 8 and 13 are directed to statutory subject matter. Appellants further assert that the Examiner must provide a reference that discloses each and every claim limitation in order to support a *prima facie* case of anticipation. M.P.E.P. §2131.

The Examiner further cites *In re Dembiczak*, 175 F.3d 994, 1000, 50 U.S.P.Q.2d 1614, 1618 (Fed. Cir. 1999) and *In re Gulack*, 703 F.2d 1381, 1385, 217 U.S.P.Q. 401, 404 (Fed. Cir. 1983). Paper No. 5, page 8. Appellants are unsure as to what proposition the Examiner is citing these cases.

The court in *In re Dembiczak*, 175 F.3d 994, 1000, 50 U.S.P.Q.2d 1614, 1618 (Fed. Cir. 1999) ruled that the Board's conclusion of obviousness, as a matter of law, cannot stand since the Examiner did not present a suggestion, teaching, or motivation to combine the prior art references cited against the pending claims. There is no language in the cited case site related to "functionality" or to "descriptiveness". Furthermore, *In re Dembiczak* deals with obviousness and not with respect to anticipation. Appellants respectfully request the Examiner to clarify the citing of *In re Dembiczak* pursuant to 37 C.F.R. §1.104(c)(2).

The court in *In re Gulack*, 703 F.2d 1381, 1385, 217 U.S.P.Q. 401, 404 (Fed. Cir. 1983) ruled that there must be *differences* between the appealed claims and the prior art in order to establish patentability. The Examiner has not proven that Teoman discloses each and every limitation of Appellants' claimed invention and hence has not established a *prima facie* case of anticipation. M.P.E.P. §2131. That is, Appellants have shown that there are differences between Appellants' claimed invention and the Examiner's cited art, Teoman. Consequently, the Examiner has not established a *prima facie* case of anticipation. M.P.E.P. §2131. If the Examiner is asserting *In re Gulack* for any other reason than described above, Appellants respectfully request the Examiner to clarify the citing of *In re Gulack* pursuant to 37 C.F.R. §1.104(c)(2).

Further, Teoman instead discloses that the user cache includes boot firmware for storing program code that is used to support operation of the user cache at system startup. Column 7, lines 50-52. Teoman further discloses that the use cache manager responds by generating I/O requests to retrieve the data identified by the preload parameters from mass storage. Column 9, lines 57-59. There is no language in the cited passages of Teoman that discloses that each of the software element objects is associated with attribute data, where the attribute data comprises one or more of an operating system information and an installation information. The Examiner has not provided any basis in fact and/or technical reasoning to support his interpretation that the cited passages disclose that each of the software element objects is associated with attribute data, where the attribute data comprises one or more of an operating system information and an installation information. *See Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990); *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Instead, the Examiner is simply relying upon his own subjective opinion which is insufficient to establish a *prima facie* case of anticipation in rejecting claims 3, 8 and 13. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

E. Claims 4, 9 and 14 are not anticipated by Teoman.

Appellants respectfully assert that Teoman does not disclose "wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number" as recited in claim 4 and similarly in claims 9 and 14. The Examiner states that attributes 111 in Figure 6 points to device driver 45 in Figure 6 and that a part number is inherent in device driver 45. Paper No. 5, page 11. Appellants respectfully traverse and assert that Teoman instead discloses a driver object, e.g., element 111 of Figure 6, that includes a pointer to a lower level driver object, e.g., element 47 of Figure 6. Column 11, lines 19-26. However, there is no language in Teoman that discloses that the attributes of a driver object include a

part number. Thus, Teoman does not disclose all of the limitations of claims 4, 9 and 14, and thus Teoman does not anticipate claims 4, 9 and 14. M.P.E.P. §2131.

Furthermore, Appellants respectfully assert the assertion that the attributes in a driver object inherently include a part number.[1] The Examiner must provide a basis in fact and/or technical reasoning to support the assertion that the attributes in a driver object inherently include a part number. *Ex parte Levy*, 17 U.S.P.Q.2d 1461, 1464 (Bd. Pat. App. & Inter. 1990). That is, the Examiner must provide extrinsic evidence that must make clear that the attributes in a driver object inherently include a part number, and that it be so recognized for persons of ordinary skill. *In re Robertson*, 169 F.3d 743, 745, 49 U.S.P.Q.2d 1949, 1950-51 (Fed. Cir. 1999). Since the Examiner has not provided such evidence, the Examiner has not presented a *prima facie* case of anticipation in rejecting claims 4, 9 and 14. M.P.E.P. § 2131.

In response to Appellants' above argument, the Examiner cites to M.P.E.P. §2106 (IV)(B)(1)(b), *In re Dembiczak* and to *In re Gulack* asserting that the above-cited limitation is non-functional[2] and then concludes that elements 15, 108, 110, 112 of Teoman disclose the limitations of claims 4, 9 and 14. Paper No. 5, page 9. Appellants respectfully traverse.

Teoman instead discloses that initially, an application process generates an I/O request to read or write a file (element 15) stored on a mass storage device. Column 5, lines 30-32. There is no language in Teoman that discloses that element 15 of Teoman is a software element object associated with attribute data, where the attribute

---

1 Appellants assume that the Examiner meant to state that the attributes in a driver object inherently include a part number instead of stating "where part number is inherent in device driver 47." Appellants do not understand the connection with a part number in device driver 47 when Figure 6 illustrates attributes in driver object 111.

2 Appellants will not again address the Examiner's argument pertaining to M.P.E.P. §2106, *In re Dembiczak* and *In re Gulack* for the sake of brevity. Appellants respectfully direct the Board's attention to Section D of Appellants' Appeal Brief regarding Appellants response to the Examiner's citation of M.P.E.P. §2106, *In re Dembiczak* and *In re Gulack*.

data comprises a part number. Thus, Teoman does not disclose all of the limitations of claims 4, 9 and 14, and thus Teoman does not anticipate claims 4, 9 and 14. M.P.E.P. §2131.

Furthermore, Teoman instead discloses that elements 108, 110 and 112 correspond to attributes in a file object and in driver objects, respectively, as illustrated in Figure 6. There is no language in Teoman that discloses that elements 108, 110 and 112 of Teoman are software element objects associated with attribute data, where the attribute data comprises a part number.

Furthermore, the Examiner asserts that Appellants' above citing of *Ex parte Levy* and *In re Robertson* are improper because the limitations of claim 4 are non-functional. Paper No. 5, page 9. Appellants respectfully traverse. The Examiner has not proven that the limitations of claim 4 are non-functional. Further, the Examiner has not cited to any authority that the Examiner does not have to provide evidence that a reference inherently discloses limitations that the Examiner deems to be non-functional. The Examiner is merely relying upon his own subjective opinion in rejecting claim 4 which is insufficient to establish a *prima facie* case of anticipation. *See In re Lee*, 61 U.S.P.Q.2d 1430, 1434 (Fed. Cir. 2002).

VIII. CONCLUSION

For the reasons noted above, the rejections of claims 1-4, 6-9 and 11-14 are in error. Appellants respectfully request reversal of the rejections and allowance of claims 1-15.

Respectfully submitted,
WINSTEAD SECHREST & MINICK P.C.
Attorneys for Appellants

By: _____
Robert A. Voigt, Jr.
Reg. No. 47,159
Kelly K. Kordzik
Reg. No. 36,571

P.O. Box 50784
Dallas, Texas 75201
(512) 370-2832

APPENDIX

1. A method for creating a preload, wherein an object of said preload is an aggregation of one or more software element objects, comprising the steps of:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object; and

installing software associated with said identified one or more software elements objects onto a particular preload associated with said particular preload object.

2. The method as recited in claim 1 further comprising the step of:

modifying an attribute of said identified one or more software element objects to match said one or more attributes of said particular preload object.

3. The method as recited in claim 1, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises one or more of an operating system information and an installation information.

4. The method as recited in claim 1, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.

5. A method for creating a preload, wherein an object of said preload is an aggregation of one or more software element objects, comprising the steps of:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object;

installing software associated with said identified one or more software elements objects onto a particular preload associated with said particular preload object;

transmitting one or more part numbers associated with said identified one or more software element objects to a manufacturing system; and

retrieving software associated with said identified one or more software element objects based on said one or more part numbers;

wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.

6. A computer program product having a computer readable medium having computer program logic recorded thereon for creating a preload, comprising the programming steps of:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object; and

installing software associated with said identified one or more software elements objects onto a particular preload associated with said particular preload object.

7. The computer program product as recited in claim 6 further comprises the programming step of:

modifying an attribute of said identified one or more software element objects to match said one or more attributes of said particular preload object.

8. The computer program product as recited in claim 6, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises one or more of an operating system information and an installation information.

9. The computer program product as recited in claim 6, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.

10. A computer program product having a computer readable medium having computer program logic recorded thereon for creating a preload, comprising the programming steps of:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object;

installing software associated with said identified one or more software elements objects onto a particular preload associated with said particular preload object;

transmitting one or more part numbers associated with said identified one or

more software element objects to a manufacturing system; and

retrieving software associated with said identified one or more software element objects based on said one or more part numbers;

wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.

11. A system, comprising:

a processor; and

a memory unit coupled to said processor, wherein said memory unit is operable for storing a computer program for creating a preload, wherein an object of said preload is an aggregation of one or more software element objects, wherein the computer program is operable for performing the following programming steps:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object; and

installing software associated with said identified one or more software element objects onto a particular preload associated with said particular preload object.

12. The system as recited in claim 11, wherein the computer program is further operable for performing the following programming step:

modifying an attribute of said identified one or more software element objects to match said one or more attributes of said particular preload object.

13. The system as recited in claim 11, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises one or more of an operating system information and an installation information.

14. The system as recited in claim 11, wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.

15. A system, comprising:

a processor; and

a memory unit coupled to said processor, wherein said memory unit is operable for storing a computer program for creating a preload, wherein an object of said preload is an aggregation of one or more software element objects, wherein the computer program is operable for performing the following programming steps:

defining a particular preload object with one or more attributes;

comparing attributes of said one or more software element objects with said one or more attributes of said particular preload object, wherein each of said one or more software element objects constitutes one or more of a device driver object, an operating system object and an application software object;

identifying one or more of said one or more software element objects whose attributes comprise said one or more attributes of said particular preload object;

installing software associated with said identified one or more software element objects onto a particular preload associated with said particular preload object;

transmitting one or more part numbers associated with said identified one or more software element objects to a manufacturing system; and

retrieving software associated with said identified one or more software element objects based on said one or more part numbers;

wherein each of said one or more software element objects is associated with attribute data, wherein said attribute data comprises a part number.